Chapter 28 KIDS Programmer Tools: Creating Builds

KIDS introduces significant revisions to the process of exporting packages over the previous export mechanism, DIFROM. For an introduction to KIDS, please see the KIDS System Management: Installations chapter.

A functional listing of the KIDS options supporting package export is:

TASK	OPTION NAME
Create Build Entry	• Create a Build using Namespace
	• Copy Build to Build
	• Edit a Build
Create a Distribution	• Transport a Distribution

The menu path for these options is:

```
Kernel Installation and Distribution System...          [XPD MAIN]
  Edits and Distribution ...               [XPD DISTRIBUTION MENU]
    Create a Build Using Namespace           [XPD BUILD NAMESPACE]
    Copy Build to Build                            [XPD COPY BUILD]
    Edit a Build                                   [XPD EDIT BUILD]
    Transport a Distribution               [XPD TRANSPORT PACKAGE]
```

This chapter covers each of these tasks, describing how to accomplish them using KIDS options.

Build Entries

KIDS stores the definition of a package in a new file, called the BUILD file. Individual entries in the BUILD file are called build entries, or builds for short. To export a package, you must first define a build entry for it in the BUILD file.

Unlike DIFROM, where you re-used the same PACKAGE file entry each time you exported a new version of a package, with KIDS you create a new BUILD file entry each time you export a package version. One advantage of having one BUILD entry per package version is that you have a complete history of each version of your package, which makes it easier to compare previous versions of a package with the current version.

After you create the build name, KIDS give you the option to choose the type of build you are creating. There are three types to choose from:

- Single
- Multi-Package
- Global

■ Choosing a Build Type Sample

```
Select Edits and Distribution Option: EDIT a Build
Select BUILD NAME: TEST 5.0
  Are you adding 'TEST 5.0' as a new BUILD (the 104TH)? Y <RET>
(Yes)
   BUILD PACKAGE FILE LINK: RET
   BUILD TYPE: SINGLE PACKAGE// ?
     Choose from:
       0        SINGLE PACKAGE
       1        MULTI-PACKAGE
       2        GLOBAL PACKAGE
   BUILD TYPE: SINGLE PACKAGE// GLOBAL <RET> GLOBAL PACKAGE
```

The following KIDS options, described below, support creating and maintaining build entries:

- Create a Build Using Namespace
- Copy Build to Build
- Edit a Build

Create a Build Using Namespace

You can quickly create a build entry and populate its components by namespace. The Create a Build Using Namespace option searches for all components in the current database matching a given list of namespaces (you can exclude by namespace also). The option searches for components of every type that match the namespace(s) and populates the build entry with all matches it finds on the system. You can then use Edit a Build to fine-tune the build entry.

As well as creating a new build entry, you can use this option to populate an existing build entry by namespace. In this case, you are asked if you want to purge the existing data. If you answer YES, the option purges the build components in the entry, and then populates the build components by namespace. If you answer NO, the option merges all components matching the selected namespaces into the existing build entry; it removes nothing already in the current build entry.

■ Kernel V. 8.0 Component Types

Print Template	Bulletin	Protocol
Sort Template	Mail Group	List Template
Input Template	Help Frame	HL7 Application Parameter
Form	Routine	HL Lower Level Protocol
Function	Option	HL Logical Link
Dialog	Security Key	Remote Procedure

■ Populating a Build Entry by Namespace

```
Select Edits and Distribution Option: Create a Build Using
Namespace

Select BUILD NAME: ZXGY 1.0
  Are you adding 'ZXGY 1.0' as a new BUILD (the 14th)? YES
   BUILD PACKAGE FILE LINK: <RET>


Namespace: ZXG
Namespace: -ZXGI
Namespace: <RET>

NAMESPACE  INCLUDE                 EXCLUDE
           -------                 -------
           ZXG                     ZXGI

OK to continue? YES// <RET>
...SORRY, LET ME THINK ABOUT THAT A MOMENT...

    ...Done.
```

■ Copying a Build Entry

```
Select Edits and Distribution Option: COPY Build to Build

Copy FROM what Package: ZXG TEST 1.0
Copy TO what Package: ZXG TEST 1.1
  ARE YOU ADDING 'ZXG TEST 1.1' AS A NEW BUILD (THE 5TH)? Y <RET>
(YES)
   BUILD PACKAGE FILE LINK: <RET>

OK to continue? YES// <RET>
...HMMM, LET ME PUT YOU ON 'HOLD' FOR A SECOND...    ...Done.
```

Copy Build to Build

You can create a new build entry based on a previous entry using the Copy Build to Build option. Note that with KIDS, you must create a new build entry for each new version of a package. This option gives you a way to quickly copy a previous build entry to a new entry. You can then use the Edit a Build to fine-tune the copied build entry.

If you choose an existing entry to copy into, the option purges the existing entry first before copying into it.

Edit a Build

Using the Edit a Build option, you can create new build entries and edit all parts of existing build entries. Edit a Build is a VA FileMan ScreenMan-driven option. There are four main screens in the Edit a Build. The following sections describe in detail each part of a build entry and how you can edit each part.

KIDS Build Checklists

KIDS Build Checklists are provided in an appendix. They are designed in conjunction with the Edit a Build option to help you plan your build entries.

■ **Functional Layout, Edit a Build**

Screen	Build Section	Build Sub-Section
Screen 1	Build Name Date Distributed Description Environment Check Routine Pre-Install Routine Post-Install Routine Pre-Transportation Routine	
Screen 2	Files and Data	Partial DD Definition Send Data Definition
Screen 3	Build Components	Print Template Sort Template Input Template Form Function Dialog Bulletin Mail Group Help Frame Routine Option Security Key Protocol List Template HL7 Application Parameter HL Lower Level Protocol HL Logical Link Remote Procedure
Screen 4	Install Questions Required Builds Package File Link Package Tracking	

Edit a Build: Name & Version, Build Information

When you invoke the Edit a Build option, KIDS loads a four-page ScreenMan form. The first screen of the form lets you edit the following package settings:

- Name
- Date Distributed
- Description
- Environment Check Routine

- Pre-Install Routine
- Post-Install Routine
- Pre-Transportation Routine

Build Name

The name of a build entry is where KIDS stores both the package's name and version number. The build name must be a package name, followed by a space and then followed by a version number. This means that every version of a package requires a separate entry in the BUILD file. One way that this is an advantage is that you have a record of the contents of every version of a package that you export.

■ Screen 1 of Edit a Build Sample

```
                          Edit a Build                        PAGE 1 OF 4
Name: ZXG TEST 1.0                      TYPE: SINGLE PACKAGE
------------------------------------------------------------------------------

                       Name: ZXG DEMO 1.0

            Date Distributed: AUG 29,1994

                 Description:

 Environment Check Routine:

        Pre-Install Routine: ZXGPRE

       Post-Install Routine: ZXGPOS

Pre-Transportation Routine:
------------------------------------------------------------------------------



COMMAND:                              Press <PF1>H for help      Insert
```

Edit a Build: Files

The second screen of Edit a Build is where you enter all the files to export with your package. For each file, you can choose whether or not to send data with the file definition.

Data Dictionary Update

The installing site is no longer asked whether they want to override data dictionary updates: data dictionary updates are determined entirely by how you, the developer, export the file. There are two settings in KIDS you can use to determine whether KIDS should update a file's data dictionary at the installing site.

If you answer YES to Update the Data Dictionary, the data dictionary will be updated at the installing site. If you answer NO, the only time the data dictionary is updated is if the file does not exist on the installing system.

You can enter M code in the Screen to Determine DD Update field. The code should set the value of $T. If $T is true, KIDS installs the data dictionary; if $T=0, KIDS does not. The screen is only executed if the data dictionary already exists on the installing system, however; if the data dictionary doesn't already exist, the file is installed unconditionally (the screen is not executed). You can use the code in this field, for example, to examine the target environment to determine whether to update a data dictionary (providing the data dictionary already exists).

Sending Security Codes

With KIDS, you can specify on a file-by-file basis whether to send security codes. For each file, you can set Send Security Code to either YES or NO.

If you answer YES to send security codes, KIDS sends the security codes of the files on the development system. KIDS only updates security codes at the installing site on new files (i.e., files that don't already exist), however. Security codes for a file are **not** updated at the installing site if the file already exists.

■ Screen 2 of Edit a Build: Selecting Files

```
                          Edit a Build                          PAGE 2 OF 4
Name: ZXG DEMO 1.0                      Type: SINGLE PACKAGE
------------------------------------------------------------------------------
                          File List (Name or Number)

     NEW PERSON

COMMAND:                              Press <PF1>H for help     Insert
```

■ Data Dictionary and Data Settings

```
                          Edit a Build                          PAGE 2 OF 4
Name: ZXG DEMO 1.0                      TYPE: SINGLE PACKAGE
------------------------------------------------------------------------------
                          File List (Name or Number)
                          DD Export Options
                      File: NEW PERSON

   Send Full or Partial DD: PARTIAL

Update the Data Dictionary: YES           Send Security Code: NO

Screen to Determine DD Update

      Data Comes With File: YES

COMMAND:                              Press <PF1>H for help      Insert
```

Sending Full or Partial Data Dictionaries

KIDS supports sending out full data dictionaries (the entire file definition), and partial data dictionaries (specified fields in a file).

Full DD (All Fields)

To send the entire data dictionary, answer FULL at the Send Full or Partial DD prompt. In this case, *all* field definitions are exported. If you are sending data, you must export the FULL data dictionary.

Partial DD (Some Fields)

If you answer PARTIAL at the "Send Full or Partial DD" prompt, KIDS lets you choose what data dictionary level(s) to export. A data dictionary level is either the file number (top level of the file) or a sub-data dictionary number of a multiple (also known as a subfile). You can export any subfile, no matter how deep (every subfile's data dictionary number will be selectable).

For each data dictionary number (level) you choose to export, you can select which fields to export at that data dictionary level:

- **If you don't specify *any* fields, *all* fields are sent.** This includes any multiple fields (subfiles) at the selected data dictionary level and all descendant subfiles.

- **If you do specify fields, only the specified fields are sent.** When you specify individual fields within a data dictionary level, however, you cannot choose any multiples at that data dictionary level.

Unlike DIFROM, KIDS does not require sending the .01 field of the file if you send a partial data dictionary. Also, KIDS does not require sending out a multiple's entire data dictionary when you export a field within a multiple; you can choose to export a specific field within a multiple, at any depth.

When you export the .01 field of a multiple (by itself or by sending the entire multiple), KIDS also sends the "parent field" of the multiple (that is, the field at the next higher level of the data dictionary holding the multiple).

Whenever you export a multiple, all "parents" of the multiple all the way up to the .01 field of the file must exist at the installing site, or else you must export all "parents" (higher data dictionary levels) yourself. Otherwise, the multiple will not be installed.

Note that certain attributes (identifiers, "ID" nodes, etc.) are considered file attributes (as opposed to field attributes), and so are sent only when you send a full DD. They aren't sent with a partial DD.

■ Data Dictionary Settings Screen

```
                          Edit a Build                    PAGE 2 OF 4
Name: ZXG DEMO 1.0                       TYPE: SINGLE PACKAGE
------------------------------------------------------------------------
                          File List (Name or Number)
                          DD Export Options
                      File: NEW PERSON

   Send Full or Partial DD: PARTIAL

Update the Data Dictionary: YES          Send Security Code: NO

Screen to Determine DD Update

      Data Comes With File: YES

COMMAND:                             Press <PF1>H for help      Insert
```

■ Partial DD: Choosing DD Levels (Top Level and Sub-File) to Send

```
                          Edit a Build                    PAGE 2 OF 4
Name: ZXG DEMO 1.0                       TYPE: SINGLE PACKAGE
------------------------------------------------------------------------
                          File List (Name or Number)
                          DD Export Options
                          Data Dictionary Number
   NEW PERSON  (File-top level)
   DMMS UNITS  (sub-file)
   ALIAS  (sub-file)
   DEFINED FORMATS FOR LM  (sub-file)

COMMAND:                             Press <PF1>H for help      Insert
```

Choosing What Data to Send with a File

When you send data, you can send all of the data in a file. But KIDS also lets you send a subset of a file's data to installing sites.

In the Screen to Select Data field, you can enter M code to screen data. The M code should set $T; if $T is set to 1, the entry is sent, and if $T is set to 0, the entry is not sent. At the moment your code for the screen is executed, the local variable Y is set to the internal entry number of the entry being screened, and the M naked indicator is set to the global level @fileroot@(Y,0). Therefore, you can use the values of Y and the naked indicator in your screen.

In the Data List field, you can select a search template. The contents of the template will be the entries that are exported.

If you choose both a screen and a search template, the screen is applied to the entries stored in the search template.

■ Settings for Sending Data

```
                          Edit a Build                        PAGE 2 OF 4
Name: ZXG DEMO 1.0                      TYPE: SINGLE PACKAGE
------------------------------------------------------------------------------
                          File List (Name or Number)
                          DD Export Options
                          Data Export Options
     Site's Data: OVERWRITE

Resolve Pointers: YES       May User Override Data Update: YES

       Data List:

  Screen to Select Data


COMMAND:                              Press <PF1>H for help      Insert
```

Determining How Data is Installed at the Receiving Site

When you send data with a file, KIDS gives you several options about how the data is sent. There are four ways KIDS can install file entries at the receiving site:

ADD ONLY IF NEW FILE	Installs data at the installing site only if this file is new to the site or if there is no data in this file at the site.
MERGE	If no matching entry is found, the incoming entry is added. When the incoming entry matches an existing entry on the system, site fields that are non-null are preserved. Only null fields in a matching site entry are overwritten by incoming values. KIDS does not send out cross-references with the data. When you merge the data, however, KIDS re-indexes and creates new cross-references. Also, when you merge the data, KIDS does *not* delete the old cross-references for that data.
OVERWRITE	If no matching entry is found, the incoming entry is added. When the incoming entry matches an existing entry on the system, site fields that are non-null are overwritten by incoming data. Values in the site's fields are preserved when the incoming field value is null, however.
REPLACE	If no matching entry is found, the incoming entry is added. When the incoming entry matches an existing entry at the top level of a file, all fields in the existing entry that are fields in the incoming data dictionary are purged; then field values for the new entry are brought in. Values in fields that aren't part of the incoming data dictionary are preserved. KIDS does not send out cross-references with the data. When you replace the data, however, KIDS re-indexes and creates new cross-references. Also, when you replace the data, KIDS deletes any old cross-references for that data. With multiples, if the .01 field of an incoming multiple matches the .01 field of an existing multiple, the existing multiple entry is completely purged, and the data from the incoming multiple replaces the current multiple entirely; values for fields in the existing multiple that aren't in the incoming data dictionary are not restored.

You can specify different settings for separate files; within a file, however, all data must be installed in one of these four ways.

You can give the installing site the choice of overriding the data update. If you set May User Override Data Update to YES, the installing site has the choice of whether to bring in data that has been sent with this file. They are not given the choice of how to install data, however (add only if new file vs. merge vs. overwrite vs. replace). If you set this field to NO, the installing site cannot override bringing in data.

How KIDS Matches Incoming Entries with Existing Entries

When KIDS installs VA FileMan data, it treats incoming entries differently depending on whether the entry is a new entry for the file **or** the incoming entry matches an existing entry in the file.

KIDS decides if an incoming entry is new or matches an existing entry by checking, in order:

1. The B index of the file or multiple, or the .01 field if there is no B index.

2. The internal entry number (ien) of the entry (if applicable).

3. The identifiers of the entry (if applicable).

First, KIDS makes a tentative match based on the B index. If there is no B index, KIDS goes through the .01 field entries of the file one-by-one looking for a match.

NOTE: The B cross-reference holds the name as a subscript. The maximum length of subscripts is defined for each operating system and is stored in the MUMPS OPERATING SYSTEM file. KIDS uses this length [for example, 63 (default) or 99] as the limit of characters to compare.

If a match (either by the B cross-reference or by the first piece of the zero node) is not found, the incoming entry is considered new and is added to the file. If a match or matches are found, two additional checks are made to determine whether any of the existing entries are a match.

KIDS next checks whether the iens of any tentatively matched entries are related. If the file has a defined .001 field, the ien is a meaningful attribute of an entry. In this case, the iens must match. If the input transform of the .01

field contains DINUM, it operates the same way as a .001 field. If the ien is meaningful, and no match is found, the incoming entry is considered new and is added to the file.

If the possibility of a match remains after checking iens, KIDS performs a final check based on identifiers.

A well-designed file uses one or more identifiers to act as key fields, so that each entry is unique with respect to name and identifiers. If identifiers exist on either the target file or the incoming data dictionary, KIDS checks the values of all such identifier fields. The value of each identifier field must be the same for the existing entry and the incoming entry to be considered a match. Only the internal value of the identifier field is checked (so if an identifier is a pointer field, problems could result). Only identifiers that have valid field numbers are used in this process.

If there is still more than one matching entry after checking .01 fields, iens, and identifiers, the lowest numbered entry in the site's file is considered a match for the incoming entry for the file. On the other hand, if no match is found after checking .01 fields, iens, and identifiers, the entry is considered new and is added to the file.

Limited Resolution of Pointers

A new feature of data export provided by KIDS is resolving pointers. For each file exported with data, you can choose whether to perform pointer resolution on that file's pointer fields (with the exception of .01 fields, identifier fields, and pointer fields pointing to other pointer fields).

KIDS does not resolve pointers for .01 fields and identifier fields in files or subfiles, nor fields that point to other pointer fields. KIDS can resolve pointers, however, for all other pointer fields in a given file or subfile.

When you don't resolve pointers, and the file being installed has pointer fields, data entries for that file are installed with whatever numerical pointer values are in the pointer fields. In which case, there is a good chance that the pointer fields no longer point to the intended entries in the pointed to file.

Resolution of pointers remedies this by exporting the free text value of the pointed-to entry. When KIDS has finished installing all files and data entries at the installing site, it begins the process of resolving pointers (if any files are set to have pointers resolved).

For each field in an entry that is a pointer field, KIDS does a lookup in the pointed to file for the free text value of the original pointed-to entry. If it finds an exact and unique match, it resolves the original pointer by storing the ien of the new matching entry in the pointer field. If it can't find an exact match, either because there are no matching entries or there are multiple

matching entries, then the pointer field is left blank, and KIDS displays an error message.

Resolution of pointers works with pointed-to entries that are themselves variable pointers. In these cases, it stores which file the pointed-to entry was pointing to, and then resolves the pointer in the appropriate target file only.

Once all pointers are resolved, KIDS re-indexes each file. Each time KIDS finishes resolving pointer fields in a given file, it re-indexes that file.

Re-Indexing Files

Once all new data has been added to all files, KIDS re-indexes the files. If any of the files have compiled cross-references, the compiled cross reference routines are rebuilt. Then, if any data was sent for a file, KIDS re-indexes ALL cross references for ALL the records in the file. Only the SET logic is executed.

Data Dictionary Cleanup

If you change the definition of a field or remove a cross-reference, you must delete the field or cross reference or otherwise clean it up on the target account during the Pre-install routine. You must completely purge the target site's data dictionary of the old field definition, even if you are re-using the same node and piece for a new field. This cleanup ensures that the data dictionary will not end up with an inconsistent structure after the installation.

You no longer need to clean up word processing fields in the data dictionary, however. Before KIDS, updated data dictionary field attributes stored in word processing fields (e.g., field description or technical description) did not completely overwrite a pre-existing attribute when installed. If the incoming value had fewer lines than the pre-existing one, the install of the data dictionary did not delete the surplus lines automatically; this deletion had to be done in the pre-install. KIDS, on the other hand, completely replaces the values of word processing fields in data dictionaries.

Edit a Build: Components

In the third screen in the Edit a Build option, you can select the components of a package to include in the build.

KIDS lets you enter an explicit list of components for each component type. You are not restricted by namespace. You can select items for each type of component simply by choosing them. Items can also be selected with the * wildcard and the – exclusion sign.

With most component types, the permissible installation actions are SEND TO SITE and DELETE AT SITE. Some component types, however, have additional installation actions available; the special cases are discussed on the following pages.

■ Kernel V. 8.0 Component Types

Print Template	Bulletin	Protocol
Sort Template	Mail Group	List Template
Input Template	Help Frame	HL7 Application Parameter
Form	Routine	HL Lower Level Protocol
Function	Option	HL Logical Link
Dialog	Security Key	Remote Procedure

■ Screen 3 of Edit a Build: Components

```
                          Edit a Build                        PAGE 3 OF 4
Name: ZXG 1.0                              TYPE: SINGLE PACKAGE
------------------------------------------------------------------------------
                          Build Components

PRINT TEMPLATE              (0)
SORT TEMPLATE               (0)
INPUT TEMPLATE              (0)
FORM                        (0)
FUNCTION                    (0)
DIALOG                      (0)
BULLETIN                    (0)
MAIL GROUP                  (0)
HELP FRAME                  (0)
ROUTINE                     (0)
OPTION                      (0)
SECURITY KEY                (0)
PROTOCOL                    (0)
LIST TEMPLATE               (0)
HL7 APPLICATION PARAMETE    (0)
HL LOWER LEVEL PROTOCOL     (0)
HL LOGICAL LINK             (0)
REMOTE PROCEDURE            (0)


COMMAND:                                Press <PF1>H for help    Insert
```

NOTE: This is an expanded view of this screen in order to show you *all* of the currently available component types. You will have to scroll through the list in order to see all of the available types.

Edit a Build: Options and Protocols

Menus and Protocols are similar to other component types, except for menus and protocols, which have more than the standard SEND TO SITE and DELETE AT SITE installation actions.

NOTE: Beginning with Kernel V. 8.0, you can no longer send out an option with an attached scheduling frequency. Scheduling of options has been moved out of the OPTION file and into the new OPTION SCHEDULING file. One advantage to this is that a developer's scheduling settings will no longer overwrite a site's scheduling settings.

To indicate to the site that an option should be scheduled regularly, you should fill in the SCHEDULING RECOMMENDED field for the option. You can enter Yes, No, or Startup. This indicates to the site whether they should regularly schedule the option or not. You should list the actual frequency you recommend in the option's description. The site can then use the TaskMan option Print Recommended for Queuing Options to list all options that developers have recommended scheduling for.

■ Option and Protocol Installation Actions

Installation Action	Description
SEND TO SITE	Menu or protocol is installed at the site; any existing version already at the site is completely purged beforehand.
DELETE AT SITE	Menu or protocol is deleted at site.
USE AS LINK FOR MENU ITEMS	Designates a menu or protocol to be used as a link. The menu or protocol is not exported to the site; instead, its name is sent so that any item you link to it as a menu item or protocol (and send) becomes a sub-item on the corresponding menu or protocol at the site. KIDS does *not* disable options and protocols which have an Action of USE AS LINK FOR MENU ITEMS.

MERGE MENU ITEMS	All fields in the menu or protocol except for items are purged and replaced by the incoming values for those fields. Any items at the site that don't match incoming items are left as is. Any items that do match incoming items are completely replaced by the incoming items. The advantage with this action is that it preserves locally added items at the site. The disadvantage is that if you have removed items, the removed items are not purged at the site.

Edit a Build: Routines

Routine selection is done based on pointers to entries in the ROUTINE file, but this file is not automatically updated when programs are saved and deleted on an M system. So before adding routines to a build entry, you should run KIDS' Update Routine File option. Be sure to update all the routines and routine namespaces that you'll need to select for your build.

When selecting routines for the build, you can select individual routines by typing in their individual names. You can select a namespace group of routines by using the * wildcard. For example, to include all routines in the namespace XQ, type in XQ*. You can exclude routines by inserting the - exclusion sign before either a single name or a wild-carded namespace. For example, to exclude all routines in the XQI namespace, type -XQI*.

For each routine, you can choose one of two actions: SEND TO SITE and DELETE AT SITE. The default action is SEND TO SITE. If you choose DELETE AT SITE, the routine will be deleted at the installing site.

Installers of KIDS packages have a choice to update routines across multiple CPUs. If they choose to do this, routines will be installed (or deleted) across all CPUs the site selects. KIDS will display various status messages while each CPU is updated. Sites cannot automatically install routines in the site's manager accounts, however; you still must instruct the site to manually install any routine that goes in the manager's account.

■ **Choosing Routines**

```
                          Edit a Build                          PAGE 3 OF 4
NAME: XU 99.0                              TYPE: SINGLE PACKAGE
------------------------------------------------------------------------------
                          BUILD COMPONENTS
                              ROUTINE

 +XQSRV4                                              SEND TO SITE
  XQSTCK                                              DELETE AT SITE
  XQT                                                 SEND TO SITE
  XQT1                                                SEND TO SITE
  XQT2                                                SEND TO SITE
  XQT3                                                SEND TO SITE
  XQT4                                                SEND TO SITE
  XQTOC                                               SEND TO SITE
  XQUSR                                               SEND TO SITE


COMMAND:                              Press <PF1>H for help    Insert
```

Edit a Build: Dialog Entries (DIALOG File)

VA FileMan, starting with V. 21.0, supports the capability for other packages to store their dialog in the VA FileMan DIALOG file. Some advantages to using the DIALOG file for user interaction include:

- Separating user interaction from other program functionality. This is a helpful step for creating GUI interfaces.
- Reusing dialog. When dialog is stored in the DIALOG file, it can be re-used.
- Easily generating package error lists. If error lists are stored in DIALOG file, there is a single point of access to print a complete list of errors.
- Implementing alternate language interfaces. Multiple language versions of a dialog can be exported; also, entries for one language's set of dialogs can be swapped with entries for another language's set of dialogs.

KIDS allows you to export entries your package maintains in the DIALOG file. Simply select which DIALOG entries you want to include in your

package, as you would for any other package component, and choose an installation action for each item (the default is SEND TO SITE, the other permissible choice is DELETE AT SITE).

For more information on using the DIALOG file, please see the *VA FileMan Programmer Manual* for V. 21.0 and later.

Edit a Build: Forms

You do not need to select which blocks to send when you send VA FileMan ScreenMan forms. You only need to select the form: KIDS sends all blocks associated with a form once you've chosen the form.

Edit a Build: Templates

When you select templates (either print, sort, or input templates), KIDS appends the file number to the name of the template. This ensures that a unique entry exists for each template (since two templates of the same name could exist for two different files).

■ Selecting Templates

```
                          Edit a Build                        PAGE 3 OF 4
Name: KERNEL 8.0T14                       TYPE: SINGLE PACKAGE
------------------------------------------------------------------------------
                          Build Components
                          PRINT TEMPLATE

 +XUSER LIST    FILE #200                              SEND TO SITE
  XUSERINQ    FILE #200                               SEND TO SITE
  XUSERVER DISPLAY    FILE #19.081                    SEND TO SITE
  XUSERVER HEADER    FILE #19.081                     SEND TO SITE
  XUUFAA    FILE #3.05                                SEND TO SITE
  XUUFAAH    FILE #3.05                               SEND TO SITE
  XUUSEROPTH    FILE #19.081                          SEND TO SITE
  XUUSEROPTP    FILE #19.081                          SEND TO SITE




COMMAND:                            Press <PF1>H for help      Insert
```

Transporting a Distribution

Once you have created a build entry and added all of the files and components you want to export, you are ready to export your package. KIDS uses a transport global as the mechanism to move data. INIT routines are no longer the transport mechanism (which removes the old restrictions on the amount of data you can export). Transport globals can then be written to distributions, which are HFS files. Use the TRANSPORT option to generate transport globals and create distributions.

Depending on how you answer the questions in this option, the transport globals this option generates can be stored:

- In a distribution, which is then ready to export as a host file.
- In a PackMan message (to be sent over the network).
- In the ^XTMP global on your local system.

If you choose to transport the distribution via a host file enter HF after the "Transport through (HF)Host File or (PM)PackMan:" prompt and enter a host file name after the "Enter a Host File" prompt. The option creates transport globals and puts them in the distribution (HFS file) that you specify.

■ Creating a Distribution

```
Select Edits and Distribution Option: Transport a Distribution

Enter the Package Names to be transported. The order in which
they are entered will be the order in which they are installed.


First Package Name: ZXG DEMO 1.0
Another Package Name: ZXG TEST 1.0
Another Package Name: <RET>

ORDER   PACKAGE
  1     ZXG DEMO 1.0
  2     ZXG TEST 1.0


OK to continue? NO// YES
Transport through (HF)Host File or (PM)PackMan: HF <RET> Host File

Enter a Host File: ZXG_EXPT.DAT
Header Comment: export of ZXG package <RET>


     ZXG DEMO 1.0...
     ZXG TEST 1.0...

Package Transported Successfully

Select Edits and Distribution Option:
```

If you don't enter a host file name, KIDS creates the transport globals and stores them in your local ^XTMP global, but does not write them to a distribution file.

If you've previously created a transport global for this package in the ^XTMP global and it still exists, KIDS asks you if you want to use what was already generated or if you want to re-generate the transport globals instead.

If you want the distribution sent via a PackMan message enter PM after the "Transport through (HF)Host File or (PM)PackMan:" prompt. You can only send *one* transport global per PackMan message, however.

■ Sending via Network (PackMan Message)

```
Select Edits and Distribution Option: TRANSport a Distribution

Enter the Package Names to be transported. The order in which
they are entered will be the order in which they are installed.


First Package Name: TEST 1.1
Another Package Name: <RET>

ORDER   PACKAGE
  1    TEST 1.1


OK to continue? NO// YES
Transport through (HF)Host File or (PM)PackMan: PM <RET>  PackMan


     TEST 1.1...
No Package File Link
Subject: TEST
Please enter description of Packman Message


TEST <RET>


 Created by DOE,JOHN at KERNEL.ISC-SF.VA.GOV  (KIDS) on MONDAY,
10/07/96 at 15:21
Do you wish to secure this message? No// ?

If you answer yes, this message will be secured to insure that
what you send is what is actually received.
Do you wish to secure this message? No// Y <RET> (Yes)
Enter the scramble hint: THIS IS A HINT <RET>
Enter scramble password:      (password is not echoed back)
Securing the message, now.  This may take a while !!!

Send mail to: DOE,JOHN     Last used MailMan: 04 Oct 96 15:28
  Select basket to send to: IN// <RET>
And send to: <RET>
```

Splitting a Distribution Across Diskettes

You can also split a distribution across diskettes. You are only offered this choice if the M implementation you are using is MSM, and the host file name you enter for the distribution starts with "A:" or "B:". If this is the case, you will be asked, by the Transport a Distribution option:

```
Size of Diskette (1K blocks):
```

You can enter a number from 0 to 1440, where 0 means unlimited size. KIDS will then split the distribution into pieces, with each piece containing the number of 1K blocks you specify (if you choose a number other than 0). KIDS will ask, between pieces:

```
Insert the next diskette and Press the return key:
```

KIDS uses the same filename for each diskette, so be sure to use a separate diskette for each piece of the distribution file. You need to label the diskettes sequentially, e.g., #1, #2, etc. The installing site will be asked to insert the diskettes by sequence number when they install the distribution.

When to Transport More than One Transport Global in a Distribution

If several packages are unrelated, they should be sent as separate distributions. This gives the installing site optimum flexibility to decide when to do each installation.

If a group of packages is to be installed together, however, and if there are dependencies between the packages, sending the packages together in one distribution can give you more control over how the group of packages is installed. If in some cases only packages A and B should be installed, and in other situations only packages A and C should be installed, and you can do the determination yourself (in each package's environment check routine), sending the group of packages in a single distribution lets you control which packages in the distribution actually are installed.

When you are using PackMan messages to send your package (rather than using a distribution), you are limited to sending only one transport global per PackMan message.

Multi-Package Builds

Multi-Package builds contain a list of other builds and lists their installation order. A Multi-Package build will transport this list of builds (template or meta-build).

■ Multi-Package Builds Sample

```
                  Edit a Build                              PAGE 1 OF 1
Name: TEST 3.0                    TYPE: MULTI-PACKAGE
------------------------------------------------------------------------

                     Name: TEST 3.0

          Date Distributed: OCT 9,1996

               Description:


Install Order          Packages or Patches
  1                    TEST 1.0
  2                    TEST 1.1




COMMAND:                              Press <PF1>H for help    Insert
```

Exporting Globals with KIDS

KIDS in Kernel V. 8.0 supports the installation of global distributions (distributions that export globals). KIDS now supports the creation of global distributions by developers. Any number of globals can be included in a build. You are given the opportunity to run an environment check before installing the global and post-install routines after installing the globals. You also are given the choice of killing globals prior to installing new globals at a site. If you answer No to this question, the global is merged with any previously installed global at the site. For more information on global distributions, see the "KIDS System Management: Installations" chapter.

■ Exporting Global Distributions Sample

```
                          Edit a Build              PAGE 1 OF 1
Name: TEST 5.0                             TYPE: GLOBAL PACKAGE
------------------------------------------------------------------

                    Name: TEST 5.0

       Date Distributed: OCT 9,1996

            Description:

 Environment Check Rtn.:          Post-Install Rtn.:

   Globals                        Kill Global Before Install?
   TMP(100)                       NO



_____________________________________________________________________



COMMAND:                      Press <PF1>H for help     Insert
```

Creating Transport Globals that Install Efficiently

There are some choices you can make when designing your build entries, to make your transport globals install efficiently at the receiving site. In particular, you can improve the efficiency of exporting data entries using KIDS:

- When exporting data, you can use the ADD IF NEW option to only add entries if the file didn't exist prior to the installation. Data is only added if the file is created by the installation. You can use this option to avoid re-exporting data for static files.

- When exporting data, send only the data you need to (KIDS no longer forces you to send all data in a file when you only need to send some of the data). You can select a subset of data to send by using a screen, a search template, or both a screen and a search template.

- When exporting data, resolve pointers only if necessary, because resolving pointers adds significant overhead to the process of loading data entries.